What Measures Do Vendors Use for Software Assurance?

Jeremy Epstein, SRI International [vita[1]][2]

2009-02-10

Books and articles frequently exhort developers to build secure software by designing security in. A few large companies (most notably Microsoft) have completely reengineered their development process to include a focus on security. However, for all except the largest vendors, software security (or software assurance) is a relatively recent phenomenon, and one with an uncertain payoff. In this article, we examine what vendors do to ensure that their products are reasonably secure. Our conclusion is that software vendors put significant energy into software security, but there is significant variation in where they invest their money.

Introduction

Concern that software products are secure has been around for more than three decades, but until relatively recently little attention was given to software security by the vendor community. The never-ending series of vulnerabilities in Microsoft software galvanized Microsoft, and resulted in their developing a security-focused life cycle [Howard 2006[3]]. Numerous other texts have described the risks of insecure software, including [Viega 2001[4]] and [McGraw 2006[5]]. More recently, an industry consortium has been formed by some of the larger software companies to define best practices for building secure software [Safecode 2008[6]].

Building on the demand, start-up companies[7] have developed tools to help identify security flaws using techniques such as source code analysis (e.g., Fortify Software, Coverity), binary code analysis (e.g., Veracode), dynamic testing (e.g., SPI Dynamics, NT Objectives, Cenzic), as well as service-focused companies that perform scheduled scans (e.g., Qualys, White Hat Security), education and engineering analysis (e.g., Aspect Security, Cigital), or penetration testing (e.g., Matasano Security).

Given the choices, vendors, especially those whose primary focus is not security, have difficulty determining where to spend their resources. Additionally, for vendors whose primary products are not security technology, there may be relatively little explicit interest from customers, thus reducing the perceived demand [Epstein 2006[8]].

In order to determine what the best practices are that we should follow, we did an informal survey of software vendors to determine how they achieve software security, what motivated them to put energy into software security, and related topics. This article presents the results of this study, along with its limitations. It does *not* make recommendations about what any particular vendor should do but rather establishes the norms as practiced at this writing.

Study Topics and Limitations

The goals of our study were to address four basic questions:

- *Who* in the organization is involved in software assurance? In particular, we wanted to know:

1. http://buildsecurityin.us-cert.gov/bsi/about_us/authors/1095-BSI.html (Epstein, Jeremy)
2. This paper describes work done while the author was employed by Software AG.
3. #dsy1093-BSI_howard
4. #dsy1093-BSI_viega
5. #dsy1093-BSI_mcgraw
6. #dsy1093-BSI_safe
7. Inclusion in the non-comprehensive list here should not be interpreted as an endorsement by the author or his employer. Some of the vendors listed offer products and/or services in addition to those in the list.
8. #dsy1093-BSI_epstein

    - Whether there is a centralized assurance person or team, or whether responsibility is distributed to each engineering team
    - Who has overall responsibility for software assurance, and where that person reports in the organization
    - Whether that person is part of the release decision process, and if so whether they have a veto (i.e., to prevent a product from being released if there are significant security flaws)
- *What* does the organization do to gain software assurance? In particular, we wanted to know whether the organization:
    - Performs threat modeling to determine the risk factors
    - Performs security design reviews to try to avoid security problems
    - Performs source code reviews (manual or automated) to find implementation flaws
    - Performs automated scans (including input fuzzing) to find implementation flaws
    - Uses penetration testing (either in-house or third-party) to search for more subtle design or implementation vulnerabilities
    - Provides developer training (and if so, how much and how frequently) so developers can avoid introducing implementation flaws
    - Has an indication (whether by gut feel or metrics) as to which technique(s) are most effective in reducing or eliminating software flaws
- *Why* does the organization have a software assurance program? For example:
    - Is the interest in software assurance due to direct customer demand, avoiding notoriety, government regulation, etc.?
    - How often do customers ask about assurance? Or do they just expect it's there?
    - What words do customers use when asking about assurance?
    - Is the organization seeing procurement language that asks about assurance?
    - Do customers or third parties (e.g., self-styled "security researchers") test the vendor's products for security?
- *When* did the organization start to focus on software assurance, and how long did it take to see results?

Our study focused exclusively on vendors of shrink-wrapped software. We deliberately eliminated several other types of companies that include software in their offerings, despite the fact that their products or services might be interesting in the context of this study:

- Custom software developers. Custom software is driven by specific customer requirements and not by the need to find the common set of capabilities that meet the common needs of a large set of customers. Thus assurance may be given more or less emphasis, depending on the particular customer. This category includes companies that primarily develop software for the government marketplace, including GOTS[9] (government off-the-shelf) software.
- Systems integrators. Similar to the custom software developers, these vendors are driven by specific customer requirements and not by the goal of offering shrink-wrapped software.
- Software as a service. While companies like Salesforce.com and WebEx.com have significant security concerns, they are not (generally) selling their software for installation on the purchaser's computers but rather *use* of that software through a hosted web site. This would be a logical area to extend the survey, as these vendors are most similar to the shrink-wrapped software market and are most at risk due to their products being publicly exposed. Our motivation for excluding these companies was primarily to avoid comparing shrink-wrap to non-shrink-wrap offerings.
- E-commerce. E-commerce vendors such as Amazon.com have significant software investments, and are at significant risk. However, software is not their primary business but rather a tool to accomplish their mission.
- Very small vendors. Unless they are specifically focused on security, there is little real motivation or ability for them to put energy into software assurance, although their products may be at risk.

- Embedded systems vendors (e.g., for medical instruments, cash registers). Because these are more likely to run in a constrained environment, and for some categories are more subject to regulation, we did not consider them a useful comparison to our environment.
- Direct competitors to the author's employer. We wouldn't expect cooperation from our competitors, as they might believe that we are gathering information to use against them.

Naturally, some companies fit in more than one category. For those, we made a decision whether to include them in our survey.

Our emphasis was on medium to large software vendors. We specifically did not seek vendors who are primarily focused on selling security products such as firewalls, IDS, and PKI, although some of those vendors are in our sample.

The list of target vendors was selected by reviewing a list of the top 500 software vendors [SWMag 2007[10]][11] and removing those who met one or more of the exclusions listed above. From the remaining list, the author focused initially on vendors where he knew one or more employees. These employees were usually, but not always, security specialists. In each case, the author asked his contacts for the name of the person or people responsible for software security. In most cases, the author was able to identify an appropriate person, and in most cases, the vendors supplied the information requested in the form of a telephone interview.

Because the author started with vendors where he had contacts, the list of targeted vendors is somewhat skewed. Most of the author's professional peers are in the security business, and he knows many people in the industry. Thus, if the author does not have any contacts in a vendor, it may be an indication that the vendor does not have a focus on security. To reduce this bias, the author reviewed lists of attendees at security conferences to identify security specialists and attempted to contact vendors through those security specialists. In some cases, targets were identified through social networks such as LinkedIn. These methods were less successful, as the personal contacts were more willing to be forthcoming than people who did not know the author and therefore had no reason to trust him.

We specifically excluded Microsoft from this survey because their security processes are well known and have been described in numerous presentations and books, especially [Howard 2006[12]]. Had we included them, their results would have shown that they use all of the techniques addressed in this article and have numerous motivations for practicing software assurance, most notably the impact on their reputation.

Study Results

Our study included eight vendors, which ranged from small (less than $100M in annual sales) to very large (more than $10B in annual sales). Sales volumes were estimated from [SWMag 2007[13]].

Vendors were classified as "security" or "non-security" depending on the predominance of their sales. This distinction was useful because companies perceived as being security vendors have a higher expectation from the marketplace—customers assume that security vendors will be less likely to have security flaws than non-security vendors.

Motivations for security assurance varied significantly, including:

- It's the right thing to do for customers.
- Avoiding being seen as "another Microsoft."[14]
- Fear of the "CNN moment" that affects stock price.
- Loss of sales due to customer concerns.

Additional details of the vendor responses will be in a forthcoming extended paper.

10. #dsy1093-BSI_swm
11. This list is admittedly dated, but for purposes of this study was adequate.
12. #dsy1093-BSI_howard
13. #dsy1093-BSI_swm

We found significant variation in the processes and motivations of the vendors studied. Not surprisingly, large vendors invest more in software assurance than small vendors, and security vendors put more emphasis on assurance than non-security vendors.

Every vendor asked to remain anonymous, so all vendors are represented by letters in Table 1 and Table 2, which summarize our key findings.

The columns in Table 1 should be interpreted as follows:

- Training: Does the vendor provide training to software developers? Is it a formal program? Informal? Focused seminars, such as describing new classes of attacks? Is training provided on a regular basis (e.g., annually)? Are there refresher sessions to remind developers?
- Design reviews: Does the vendor perform security-focused design reviews of software as part of the normal development process? If so, are they formal and mandatory? Are they performed by developers or security experts?
- Pentesting: Is penetration testing performed as part of the software life cycle? If so, is it done by employees of the vendor (internal), outside experts (external), staff in the field who are not part of the product development organization, by customers, or (involuntarily from the vendor's perspective) by hackers who look for and publicize problems?
- Source analysis: Is static source code analysis performed as part of the software life cycle? If so, is it done manually by reading code, or using automated tools, or both? If tools are used, are they commercial tools (such as those offered by vendors like Fortify and Coverity), simpler tools (such as the UNIX *lint* utility), or a custom-developed tool? How widely are they used in the organization—is it per product or company wide?
- Dynamic testing: Is dynamic testing done using tools such as IBM Watchfire or HP WebInspect (formerly SPI Dynamics)?

Table 1. Techniques used for assurance

Vendor	*Training?*	*Design reviews?*	*Pentesting?*	*Source analysis?*	*Dynamic testing?*
M	Informal	Informal	Internal & external	Manual	Yes
W	Formal & refresher	Not a focus	Internal, external, & customers	Proprietary tools	Yes
F	Informal & seminars	Performed by developers	Extensive internal, some external	Manual & proprietary tools	Yes
H	Formal	Informal	Internal, external & customers	Company-wide automated	Yes
B	Formal, extensive	Workshop with experts	Internal but discouraged	Company-wide automated	Yes
S	Seminars	Workshop with experts	Field only	Manual, simple tools	Minimal
K	Formal, mandatory	Performed by security expert	Varies by product	Varies by product; some automated	Yes
R	Minimal	Minimal	Not internally, but regular target by hackers	Primary focus	Minimal

The columns in Table 2 should be interpreted as follows:

- Customer expectations: Are customer expectations the primary or secondary reason for investing in software security? All vendors said it was at least a secondary motivation.
- Fear of publicity: Is fear of publicity (i.e., appearing in the media as the cause of a security failure) a primary, secondary, or minor consideration?
- Explicit requests: Are explicit requests from customers a primary or minor consideration (no vendor said it was a secondary consideration). Several vendors said the only explicit requests for software security come from government customers.

Table 2. Motivations for investments

Vendor	*Customer expectations*	*Fear of publicity*	*Explicit requests*
M	Primary	Secondary	Minor
W	Primary	Minor	Govt customers only
F	Primary	Secondary	Minor
H	Secondary	Primary	Govt customers only
B	Secondary	Minor	Primary
S	Secondary	Primary	Minor
K	Primary	Secondary	Minor
R	Primary	Minor	Govt customers only

From this limited survey, we conclude that:

- Software vendors are aware of the risks of insecure software and are generally motivated by fear of bad publicity to minimize the security vulnerabilities in their products.
- Few non-government customers explicitly ask for software assurance, but vendors believe that it’s an unspoken expectation.
- The techniques used to gain assurance vary among vendors, but nearly all agree that developer training is one of the most valuable uses of limited resources. While everyone agrees that penetration testing has its limitations, it is still helpful as a way to know how good or bad a product is.
- Source code analysis is still early in the acceptance phase, both because tools are expensive and because they are difficult to use effectively. Dynamic testing, including fuzzing, seems to be more cost-effective.
- Common Criteria was mentioned by nearly all vendors, and all but one felt it was a paperwork exercise that had almost no impact on the assurance of their products [CC[17]].
- Most organizations started focusing on software assurance several years ago, and took several years to see results.

Security engineers frequently ask why vendors sell software that has significant security problems. This survey is a step toward answering that question—customers rarely ask about assurance, but despite that, vendors are making significant strides in improving the assurance of their software.

Conclusions

Vendors are motivated by customer demand and profit. Thus far, vendors do not see profit in improved software assurance, and explicit customer demand has been minimal. Therefore, they invest primarily because of fear of bad publicity and the notion that assurance is the right thing to do.

Having noted that limitation, vendors are investing in the areas where they perceive the greatest effectiveness: developer training, penetration testing, and dynamic (black-box) testing, with a smaller level of investment in source code analysis.

Changing the level of investment and types of investment will require a substantial change in customer behavior: customers explicitly demanding assurance rather than assuming it's already done.

Acknowledgments

The author thanks his contacts in each of the vendors. As each of the vendors provided information about their processes on a non-attribution basis, he regrets that he is unable to thank them by name.

References

[CC]	*Common Criteria for Information Technology Security Evaluation*, ISO/IEC 15408.
[Epstein 2006]	J. Epstein, S. Matsumoto, and G. McGraw. "Software Security and SOA: Danger Will Robinson!" *IEEE Security & Privacy*, February 2006.
[Howard 2006]	Michael Howard and Steve Lipner. *The Security Development Lifecycle*. Microsoft Press, 2006.
[McGraw 2006]	Gary McGraw. *Software Security: Building Security In*. Addison-Wesley, 2006.
[Safecode 2008]	*Software Assurance: An Overview of Current Industry Best Practices*, February 2008, www.safecode.org[18]
[SWMag 2007]	"The Complete Searchable 2007 Software 500 Database," *Software Magazine*. http://www.softwaremag.com/S_FocusAreas.cfm?Doc=The500
[Viega 2001]	John Viega and Gary McGraw. *Building Secure Software: How to Avoid Security Problems the Right Way*. Addison-Wesley, 2001.

1. mailto:permission@sei.cmu.edu